Malware Fraud Detection In Web Application Using Content Integrity Verification

Pasupuleti Vinod[1], P. Nageswara Rao[2], Bullarao Domathoti[3]
[1]Dept of CSE
[2, 3]Associate Professor Dept of CSE
[1, 2] Swetha Institute of Technology & Science

Abstract- *Fraudulent behaviors in Google Play, the most popular Android app market, fuel search rank abuse and malware proliferation. To identify malware, previous work has focused on app executable and permission analysis. In this paper, we introduce FairPlay, a novel system that discovers and leverages traces left behind by fraudsters, to detect both malware and apps subjected to search rank fraud. FairPlay correlates review activities and uniquely combines detected review relations with linguistic and behavioral signals gleaned from Google Play app data (87K apps, 2.9M reviews, and 2.4M reviewers, collected over half a year), in order to identify suspicious apps. FairPlay achieves over 95% accuracy in classifying gold standard datasets of malware, fraudulent and legitimate apps. We show that 75% of the identified malware apps engage in search rank fraud. FairPlay discovers hundreds off fraudulent apps that currently evade Google Bouncer's detection technology. FairPlay also helped the discovery of more than 1,000 reviews, reported for 193 apps that reveal a new type of "coercive" review campaign: users are harassed into writing positive reviews, and install and review other apps.*

Keywords- Android market, search rank fraud, malware detection

I. INTRODUCTION

Google Play (previously Android Market) is a digital distribution service operated and developed by Google. It serves as the official app store for the Android operating system, allowing users to browse and download applications developed with the Android software development kit (SDK) and published through Google. Google Play also serves as a digital media store, offering music, magazines, books, movies, and television programs. It previously offered Google hardware devices for purchase until the introduction of a separate online hardware retailer, Google Store, on March 11, 2015. Applications are available through Google Play either free of charge or at a cost. They can be downloaded directly on an Android device through the Play Store mobile app or by deploying the application to a device from the Google Play website. Applications exploiting hardware capabilities of a device can be targeted to users of devices with specific hardware components, such as a motion sensor (for motion-dependent games) or a front-facing camera (for online video calling). The Google Play store had over 82 billion app downloads in 2016 and has reached over 3.5 million apps published in 2017. It has been the subject of multiple issues concerning security, in which malicious software has been approved and uploaded to the store and downloaded by users, with varying degrees of severity.

Google Play was launched on March 6, 2012, bringing together the Android Market, Google Music, and the Google eBook store under one brand, marking a shift in Google's digital distribution strategy. The services operating under the Google Play banner are: Google Play Books, Google Play Games, Google Play Movies & TV, Google Play Music, Google Play Newsstand, and Google Play Console. Following their re-branding, Google has gradually expanded the geographical support for each of the services.

II. EXISTING SYSTEM

- Google Play uses the Bouncer system to remove malware. However, out of the 7, 756 Google Play apps we analyzed using Virus Total, 12% (948) were flagged by at least one anti-virus tool and 2% (150) were identified as malware by at least 10 tools.
- Sarma et al. use risk signals extracted from app permissions, e.g., rare critical permissions (RCP) and rare pairs of critical permissions (RPCP), to train SVM and inform users of the risks vs. benefits tradeoffs of apps.
- Peng et al. propose a score to measure the risk of apps, based on probabilistic generative models such as Naive Bayes.
- Yerima et al. also use features extracted from app permissions, API calls and commands extracted from the app executables.

III. PROPOSED SYSTEM

- We propose Fair Play, a system that leverages to efficiently detect Google Play fraud and malware.

Our major contributions are:

- To detect fraud and malware, we propose and generate relational, behavioral and linguistic features, that we use to train supervised learning algorithms
- We formulate the notion of *co-review graphs* to model reviewing relations between users.
- We develop PCF, an efficient algorithm to identify temporally constrained, co-review pseudo- cliques — formed by reviewers with substantially overlapping co-reviewing activities across short time windows.
- We use temporal dimensions of review post times to identify suspicious review spikes received by apps; we show that to compensate for a negative review, for an app that has rating R, a fraudster needs to post at least positive reviews. We also identify apps with "unbalanced" review, rating and install counts, as well as apps with permission request ramps.
- We use linguistic and behavioral information to (i) detect genuine reviews from which we then (ii)extract user-identified fraud and malware indicators.

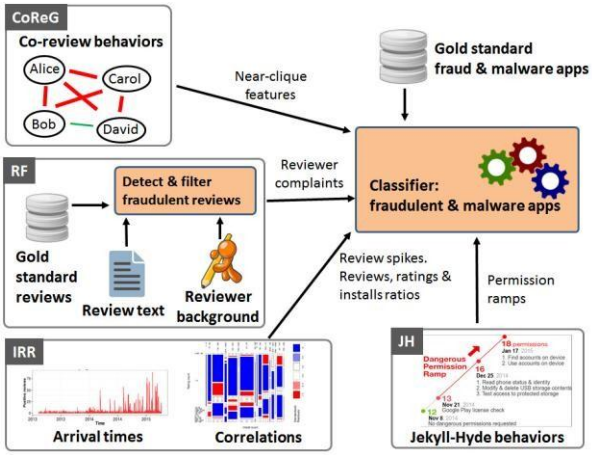

Fig 1: Fair Play system architectur

FairPlay:

Fair Play organizes the analysis of longitudinal app data into the following 4 modules, illustrated in

Figure 1. The Co- Review Graph (CoReG) module identifies apps reviewed in a contiguous time window by groups of users with significantly overlapping review histories. The Review Feedback (RF) module exploits feedback left by genuine reviewers, while the Inter Review Relation (IRR) module leverages relations between reviews, ratings and install counts. The Jekyll-Hyde (JH) module monitors app permissions, with a focus on dangerous ones.

The Co-Review Graph (CoReG) Module:

This module exploits the observation that fraudsters who control many accounts will re -use them acrossmultiple jobs. Its goal is then to detect sub-sets of an app's reviewers that have performed significant common review activities in the past. In the following, we describe the co-review graph concept, formally present the weighted maximal clique enumeration problem, then introduce an efficient heuristic that leverages natural limitations in the behaviors of fraudsters. Co-review graphs. Let the co-review graph of an app, see Figure 8, be a graph where nodes correspond to user accounts who reviewed the app, and undirected edges have a weight that indicates the number of apps reviewed in c ommon by the edge's endpoint users.

Pseudo Clique Finder (PCF):

The problem of finding dense structures in a given graph is quite basic in informatics including datamining and data engineering. Clique is a popular model torepresent dense structures, and widely used because of itssimplicity and ease in handling. Pseudo cliques are naturalextension of cliques which are subgraphs obtained by removing small number of edges from cliques. We here define a pseudo clique by a subgraph such that the ratio of the number of its edges compared to that of the clique with the same number of vertices is no less than a given threshold value. In this paper, we address the problem of enumerating all pseudo cliques for a given graph and a threshold value. We first show tha t it seems to be difficult to obtain polynomial time algorithms using straightforward divide and conquer approaches. Then, we propose a polynomial time, polynomial delay in precise, algorithm based on reverse search. The time complexity for each pseudo clique is $O(\Delta \log |V|+\min \{\Delta\ 2,|V|+|E|\})$. Computational experiments show the efficiency of our algorithm for both randomly generated graphs and practical graph

Algorithm 1 PCF algorithm pseudo-code.

```
Input: days, an array of daily reviews, and
       θ, the weighted threshold density
Output: allCliques, set of all detected pseudo-cliques
1. for d :=0  d < days.size(); d++
2.    Graph PC := new Graph();
3.    bestNearClique(PC, days[d]);
4.    c := 1; n := PC.size();
5.    for nd := d+1; d < days.size() & c = 1; d++
6.      bestNearClique(PC, days[nd]);
7.      c := (PC.size() > n); endfor
8.    if (PC.size() > 2)
9.      allCliques := allCliques.add(PC); fi endfor
10. return
11. function bestNearClique(Graph PC, Set revs)
12.   if (PC.size() = 0)
13.     for root := 0; root < revs.size(); root++
14.       Graph candClique := new Graph ();
15.       candClique.addNode (revs[root].getUser());
16.       do candNode := getMaxDensityGain(revs);
17.         if (density(candClique ∪ {candNode}) ≥ θ))
18.           candClique.addNode(candNode); fi
19.       while (candNode != null);
20.       if (candClique.density() > maxRho)
21.           maxRho := candClique.density();
22.           PC := candClique; fi endfor
23.   else if (PC.size() > 0)
24.       do candNode := getMaxDensityGain(revs);
25.         if (density(candClique ∪ candNode) ≥ θ))
26.           PC.addNode(candNode); fi
27.       while (candNode != null);
28. return
```

Reviewer Feedback (RF) Module:

Reviews written by genuine users of malware and fraudulent apps may describe negative experiences.

The RF module exploits this observation through a two step approach:

(i) detect and filter out fraudulent reviews, then
(ii) identify malware and fraud indicative feedback from the remaining reviews.

Reviewer feedback extraction. We conjecture that (i) since no app is perfect, a "balanced" review that contains both app positive and negative sentiments is more likely to be genuine, and (ii) there should exist a relation between the review's dominating sentiment and its rating. Thus, after filtering out fraudulent reviews, we extract feedback from the remaining reviews.

IV. CONCLUSION

We have introduced FairPlay, a system to detect both fraudulent and malware Google Play apps. Our experiments on a newly contributed longitudinal app dataset, have shown that a high percentage of malware is involved in search rank fraud; both are accurately identified by FairPlay. In addition, we showed FairPlay's ability to discover hundreds of apps that evade Google Play's detection technology, including a new type of coercive fraud attack.

REFERENCES

[1] Google Play. https://play.google.com/.
[2] Ezra Siegel. Fake Reviews in Google Play and Apple App Store. Appentive, 2014
[3] Stephanie Mlot. Top Android App a Scam, Pulled From Google Play. PCMag, 2014
[4] Daniel Roberts. How to spot fake apps on the Google Play store. Fortune, 2015
[5] Michael Grace, Yajin Zhou, Qiang Zhang, Shihong Zou, and Xuxian Jiang. Riskranker: Scalable and Accurate Zero -day Android Malware Detection. In Proceedings of ACM MobiSys, 2012.
[6] Iker Burguera, Urko Zurutuza, and Simin Nadjm-Tehrani. Crowdroid: Behavior-Based Malware Detection System for Android. In Proceedings of ACM SPSM, pages 15–26. ACM, 2011
[7] VirusTotal - Free Online Virus, Malware and URL Scanner. https: //www.virustotal.com/, Last accessed on May2015.
[8] Gang Wang, Christo Wilson, Xiaohan Zhao, Yibo Zhu, Manish Mohanlal, Haitao Zheng, and Ben Y. Zhao. Serf and Turf: Crowdturfing for Fun and Profit. In Proceedings of ACM WWW. ACM, 2012
[9] Jon Oberheide and Charlie Miller. Dissecting the Android Bouncer. SummerCon2012, New York, 2012.